

## ISD Detailed Design

**Number:** 580-SP-042-01  
**Effective Date:** February 18, 2006  
**Expiration Date:** February 18, 2010

**Approved By:**  
**Name:** Barbara Pfarr  
**Title:** Assoc. Chief, ISD

---

<b>Responsible Office:</b> 580/Information Systems Division (ISD)	<b>Asset Type:</b> Sub-process
<b>Title:</b> ISD Detailed Design	<b>PAL Number:</b> 2.3.2

---

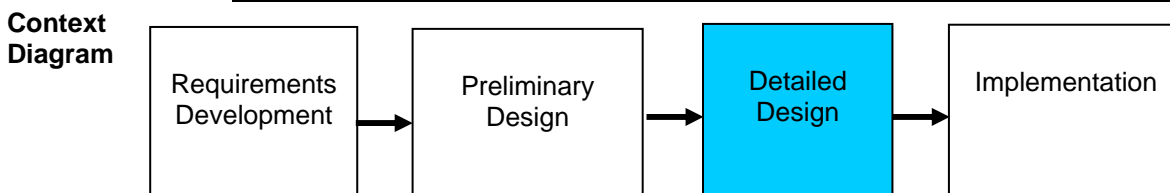
---

<b>Purpose</b>	The purpose of this Detailed Design process is for building upon the preliminary design to provide the product capabilities and architecture necessary for implementation.
----------------	--

---

<b>Scope</b>	This process is applicable to the design phase of all ISD mission software projects.
--------------	--

---



---

<b>Roles and Responsibilities</b>	<b>Product Development Lead (PDL):</b> Initiates the process.
	<b>Development Team Lead(s) (DTL):</b> Supports the PDL in executing this process.

---

<b>Usage Scenarios</b>	<p>This process is used to fully establish a product or products.</p> <p>This process may be re-entered and it's products effected because of Critical Design Review (CDR) action items/ Review Item Dispositions (RIDs).</p> <p>This process may be re-entered upon changes to requirements and this process may be re-entered iteratively depending upon the development model but iterations should not be significant enough to change the Preliminary Design and architecture.</p>
------------------------	---

---

<b>Inputs</b>	<p>Inputs include:</p> <ul style="list-style-type: none"><li>• Preliminary software design</li><li>• Validated and project-approved requirements</li><li>• Operations scenarios</li><li>• Interface documents</li><li>• Re-use plans</li><li>• Test Plan</li></ul>
---------------	--

---

- Requirements Traceability Matrix
- RIDs/RFAs from reviews

*GUIDANCE: This process connects with the Project Monitoring and Control, Requirements, Risk and Testing processes and may receive input from them.*

#### Entry Criteria

This process is entered when the preliminary software design has been approved.

*GUIDANCE: This may be achieved when all/sufficient Preliminary Design Review action items are closed, or (depending on the lifecycle model) when the next iteration is scheduled to start.*

#### Exit Criteria

All outputs are completed.

#### Output

Outputs include:

- Design Documentation
- Requirements Change Requests
- CDR Presentation Materials
- Updated Requirements Traceability Matrix
- Lessons Learned
- Suggested refined estimates of system size, effort, and schedule
- RIDs/RFAs from reviews collected and placed under appropriate configuration control for tracking to closure

*GUIDANCE: See the Process Asset Library (PAL) for further guidance and templates for Traceability Matrix, Preliminary Design Review and Lessons Learned.*

#### Major Tasks

The PDL shall perform sequentially, concurrently, and/or iteratively as necessary:

1. Expand and Refine Architecture
2. Design Software Unit(s)
3. Prepare materials

*GUIDANCE: This is an iterative process where individual task may be performed concurrently and/or out of sequence*

*GUIDANCE: This process connects with the Project Monitoring and Control, Requirements, CDR, and Testing processes and may provide input to updated: Risks, CM Plan, WBS, Build Plan, QA Needs, Testing Plans, and Test Cases*

#### Task 1:

##### Expand and Refine Architecture

- a. Decompose functions -- Considering the operating system requirements for structure of application programs and communication between application programs. Decompose the designated processing into lower-level component processing.
- b. Identify lower-level reusable software from prior efforts -- identify any reusable components that can be incorporated into the design according

---

to the reuse strategy specified in the Software Management Plan (SMP).

*GUIDANCE: The major or high-level reuse components are defined during the preliminary design process. Also identify any potential future reuse of newly created design components which may be future candidates for reuse and ensure that they are designed and packaged in such a way as to promote their reuse.*

- c. Identify software units -- Identify the software unit names. Follow the naming conventions defined for the project.
- d. Identify software unit interactions/interfaces -- Define the software unit interface requirements for all software units in the system.
- e. Select IT Security components (if applicable) -- Identify components of the CSCI which could be vulnerable to a breach of system security. Develop a security assurance strategy to ensure that the design for the identified software components minimizes or eliminates the potential for breaches of system security.
- f. Establish failure detection and correction (FDC) -- Identify components of the CSCI which could fail. Develop a correction/recovery strategy to ensure that the design for the identified software minimizes or eliminates the potential for failures of the system.

*GUIDANCE: Review any software safety requirements. If such requirements exist, develop a safety assurance strategy to minimize or eliminate the potential hazards.*

*GUIDANCE: Consider a self or peer review/walkthrough using the Design Inspection/Walkthrough Checklist (available in the PAL) at this point. For missions, Peer Reviews for major software subsystems are required.*

---

## Task 2:

### Design Software Units

*GUIDANCE: Consider operations concepts/scenarios through-out.*

- a. Establish I/O for each unit -- Generate and coordinate data and control definitions for the software unit inputs and outputs.
- b. Select Algorithms -- For each software unit (or operation on each class) select algorithms to accomplish the required function. Develop this internal logic in accordance with the standards specified in the SMP.
- c. Select Data Structures -- For each software unit (or operation on each class) select appropriate data structures to accomplish the required function.
- d. Define unit level data requirements/ communications protocols -- Define unit-level data requirements and communication protocols and formats between each software unit in the functional group.

*GUIDANCE: Consider equipment configuration, operating system*

---

*constraints, etc.*

- e. Determine “reusability” requirements for each unit -- When future reuse is an objective for the software being developed, determine applicable design requirements to facilitate reuse.

- f. Develop software unit design for each unit

**GUIDANCE:**

- *Each software unit should be limited to the logic required to accomplish a single function.*
- *Knowledge of a particular data structure should be isolated to a single software unit.*

- g. Finalize user interface(s).

**GUIDANCE:** *For flight software this will be command-and-telemetry definitions.*

- h. Estimate utilization and size of each CSCI or unit as appropriate -- Estimate resource utilization for the component of software being designed. Include CPU throughput, memory utilization, and I/O channel usage.

**GUIDANCE:** *The flight software resource usage should be evaluated against requirements and other constraints. The usual goal is 40% margin at this stage of software design (See GSFC-STD-1000 for specifics).*

- i. Conduct one or more design walkthrough/inspections so that each unit is inspected (and iterate as necessary)

**GUIDANCE:** *Use the Design Inspection/Walkthrough Checklist available in the PAL*

**GUIDANCE:** *Ensure the design is compliant with the operations concepts/scenarios.*

**GUIDANCE:** *Consider not just how the software will work, but how it will handle anomalies, fail gracefully and permit recovery.*

---

### Task 3:

#### Prepare materials

- a. Critical design review presentation materials

**GUIDANCE:** *See CDR checklist in the ISD PAL for further information. The Project Monitoring and Control process is used to track action items and feedback from the CDR – which may result in further actions in this process and refinement/update to the design.*

- b. Traceability Matrix updates

- c. Planning Refinements/Updates

**GUIDANCE:** *As a result of more detailed knowledge of the system to be developed, the Test Plan and Product Plan (including cost and schedule estimates) may be refined or significantly changed.*

- d. Lessons Learned

**GUIDANCE:** *Consider conducting a self evaluation using the Design*

*Inspection/Walkthrough checklist, and a peer evaluation using the Design Inspection/Walkthrough checklist (in the PAL) prior to preparing CDR materials.*

*See the PAL for further guidance and templates for Traceability Matrix, CDR and Lessons Learned.*

## Measures

Recommended Measures:

- Actual effort spent (cost and schedule) on the detailed design process
- Number of change requests generated

## Tools and Templates

Name	Description
Design Specification Guidelines	ISD/Code 580 – To be developed
FSW Design Guidelines Template	FSW/Code 582 – To be developed
<a href="#">Software Contents of the Mission-Level Critical Design Review</a>	ISD/580 Checklist
FSW Critical Design Review Standard	FSW/Code 582 – To be developed
FSW Contents of System Critical Design Review	FSW/Code 582 – To be developed
<a href="#">Contents of the Software Critical Design Review</a>	ISD/580 Checklist
Review Item Disposition (RID) Form	ISD/580 Form – To be developed
<a href="#">FSB Request for Action (RFA) Form</a>	FSW/Code 582 Template

## Training

Course Name	Description
<a href="#">Software Project Management</a>	Week long project management class for DTLs and PDLs. Course ID HQ0005

## References

- **Glossary:** <http://software.gsfc.nasa.gov/glossary.cfm>  
Defines common terms used in ISD processes
- **Process Asset Library:** <http://software.gsfc.nasa.gov/process.cfm>  
Library of all ISD process descriptions
- GSFC-STD-1000 – GSFC Rules for the Design, Development, Verification, and Operation of Flight Systems (“Golden Rules”), Section 3.07 “Flight Software Margin Warnings.
- Page-Jones, M., The Practical Guide to Structured Systems Design, second edition, Yourdon Press (Prentice-Hall), 1988.
- Ward, P. T. and Mellor, S. J., Structured Development for Real-Time Systems, 3 volumes, Yourdon Press, 1985, 1986.
- Yourdon, E. and Constantine, L., Structured Design, Prentice-Hall, 1979.
- Software Verification and Validation: A Practitioner's Guide by Steven R. Rakitin Artech House © 1997

- 
- Booch, Grady, Object-Oriented Design: With Applications, Benjamin/Cummings, Redwood City, CA, 1991.
  - What Makes a Good Object-Oriented Design (<http://ootips.org/ood-principles.html>)

**Quality  
Management  
System Records**

<b>Controlled Document / Description</b>	<b>Record Custodian</b>
Detailed Design document(s)	CMO
Unit design	DTL(S)
CDR Presentation Materials	PDL

**Change History**

<b>Version</b>	<b>Date</b>	<b>Description of Improvements</b>
1.0	12/23/05	Initial approved version by CCB